Information

Memory limit

The limit is 256 MiB for each problem.

Source code limit

The size of each solution source code can't exceed 256 KiB.

Submissions limit

You can submit at most 50 solutions for each problem.

You can submit a solution to each task at most once per 30 seconds. This restriction does not apply in the last 15 minutes of the contest round.

Scoring

Each problem consists of several subtasks. The subtask score is awarded if all tests in the subtask are passed.

The number of points scored for the problem is the total number of points scored on each of its subtasks. The score for the subtask is the maximum number of points earned for this subtask among all the solutions submitted.

Feedback

To get feedback for your solution, go to “Runs” tab in PCMS2 Web Client and use “View Feedback” link. In each problem of the contest you will see the score for each subtask, or the verdict for the first failed test.

Scoreboard

The contestants' scoreboard is available during the contest. Use “Monitor” link in PCMS2 Web Client to access the scoreboard. The standings provided in PCMS2 Web Client are not final.

Problem A. Enchanted Cat

Time limit: 1 second

The cat Miston is interested in non-negative numbers with n digits and is fascinated by powers of two. Help the cat determine how many n-digit numbers are divisible by 2^x without a remainder.

Input

The first line contains an integer n — the number of digits in the interesting numbers ($1 \le n \le 18$).

The second line contains an integer x — the exponent in the expression 2^x ($1 \le x \le 60$).

Output

Output a single number — the sought quantity.

Scoring

Subtask	Points	Additional constraints	Required groups	Comment
0	0	–	–	Tests from the statement.
1	10	$x = 1$	–	–
2	10	$x \le 2$	1	–
3	10	$x \le 3$	0–2	–
4	20	$n \le 6$; $x \le 20$	0	–
5	25	$x < n$	–	–
6	10	$x \ge 40$	0, 4	–
7	15	–	0–6	–

Example

standard input	standard output
3 3	112

Explanation

The suitable numbers are $104, 112, \ldots, 992$.

Problem B. SuperSavings

Time limit: 3 seconds

After being fired from the accounting department, Leviathan created a financial organization called "SuperSavings".

"SuperSavings"has n partners numbered from 1 to n.

Partner number i is characterized by two parameters—greed level a_i and patience level b_i.

Each partner can perform the operation `"Payout"`. When partner number i uses the operation, the greed level of that partner and the patience levels of all other partners decrease by 1. However, the operation is blocked by "SuperSavings"and is not executed if, after it, any partner's greed level or patience level becomes **less than** 0.

Partners perform the `"Payout"` operations in any order. One partner can perform an unlimited number of operations.

Help "SuperSavings"determine the maximum number of `"Payout"` operations that the partners can collectively perform. Blocked operations do not count.

Input

The first line contains a single integer n ($2 \le n \le 10^5$) — the number of partners in "SuperSavings".

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$).

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \le b_i \le 10^9$).

Output

Output a single integer — the maximum number of `"Payout"` operations that the partners can collectively perform.

Scoring

Group	Points	Additional constraints			Required Groups	Comments
		n	a_i	b_i		
0	0	–	–	–	–	Tests from the statement
1	19	$n \le 7$	$a_i \le 7$	$b_i \le 7$	–	
2	16	$n = 2$	–	–	–	
3	25	$n \le 100$	$a_i \le 100$	–	1	
4	20	$n \le 1000$	–	–	0, 1, 2, 3	
5	20	–	–	–	0, 1, 2, 3, 4	

Examples

standard input	standard output
4 1 2 2 3 4 2 5 4	4
2 10 10 14 2	12
4 9 9 9 9 0 0 9 9	0
9 1 2 3 1 2 3 1 2 3 9 8 7 9 8 7 9 8 7	9
2 1000000000 0 0 1000000000	1000000000
3 0 0 4 7 6 1	4

Note

In the first input set, 4 operations could be performed by partners numbered 2, 3, 2, 3. It can be shown that performing 5 operations is impossible.

In the second input set, partner number 1 could perform 2 operations, while partner number 2 could perform 10 operations.

In the third input set, the patience levels of the first two partners are zero, so it is impossible to perform any operations.

In the fourth input set, partners 2, 5, and 8 could perform one operation each, while partners 3, 6, and 9 could perform two operations each. In total: 9 operations.

In the fifth example, partner number 1 could perform all 10^9 operations.

Problem C. To School Through the Snow

Time limit: 1.8 seconds

Grandfather told how he walked to school.

Grandfather would leave home and then pass through several transitions between intersections. The intersections are connected by directed streets and directed corridors inside buildings, and movement through these transitions is only allowed in one specific direction. Outside, one can freeze (lose heat), while inside a building, one can warm up (gain heat). Grandfather reached school as quickly as possible, but he never got too cold or too hot on the way, meaning the amount of heat was always within the range of -30 to $+30$ (inclusive). At the beginning of his journey, Grandfather's heat level was always zero.

Grandfather does not remember the specific path, but he recalls n intersections numbered from 1 to n that he could traverse, and m transitions between them. For each transition (which can be either a street or a corridor), Grandfather remembers which intersection it led from and to, how long it took to traverse, and how much heat he lost (in the case of a street) or gained (in the case of a corridor) while passing through it.

Help Grandfather remember the minimum time it took him to get to school, or tell him that he forgot something and that it is impossible to get from home to school without getting too cold or too hot given the specified set of intersections and transitions.

Input

Each test consists of several sets of input data. The first line contains a single integer t ($1 \le t \le 10\,000$) — the number of input data sets. The description of the input data sets follows.

The first line of each input data set contains two integers n, m ($1 \le n, m \le 10^5$) — the number of intersections and the number of transitions between them.

The i-th of the following m lines contains 4 integers u, v, l, and dt ($1 \le u, v \le n, u \ne v$, $1 \le l \le 10^6$, $-30 \le dt \le 30$), which indicate the existence of a transition that Grandfather could traverse in time l from intersection u to intersection v with a change in heat of dt while passing through it. If $dt < 0$, then this transition is a street, and Grandfather lost $-dt$ heat while passing through it; otherwise, this transition is a corridor in a building, and Grandfather gained dt heat while passing through it.

Upon leaving home, Grandfather instantly arrived at intersection number 1. He instantly entered the school as soon as he reached intersection number n.

It is guaranteed that the sum of n (and the sum of m) across all input data sets does not exceed 10^5.

It is possible that there are multiple transitions from some intersection u to some intersection v, or that there exists both a transition from u to v and a transition from v to u.

Output

For each test, output a single number — the minimum time it took Grandfather to get from home to school. If it is impossible to reach school without getting too cold or too hot, output -1.

Scoring

The tests for this problem consist of six groups. Points for each group are awarded only if all tests in the group and all tests in the necessary groups are passed.

Subtask	Points	Additional Constraints	Necessary Groups	Comments
0	0	–	–	Tests from the statement.
1	13	$l = 1, dt = 0$	–	–
2	14	$dt = 0$	1	–
3	19	$dt > 0$	–	–
4	23	The graph is acyclic†	–	–
5	31	–	1–4	–

† The graph is acyclic, meaning that in subgroup 4, it is impossible to reach intersection u from itself (for any $1 \le u \le n$) in any non-zero number of transitions. This statement holds true even if the temperature changes on all transitions are considered to be zero.

Examples

standard input	standard output
1 5 6 1 2 2 0 1 4 4 0 2 3 3 0 2 5 1 0 3 2 4 0 4 5 2 0	3
1 5 6 1 2 2 -20 1 4 4 26 2 3 3 5 2 5 1 -15 3 2 4 10 4 5 2 27	10

Note

Illustration for the second example. In this example, the shortest path would look like this: $1 \to 2 \to 3 \to 2 \to 5$.

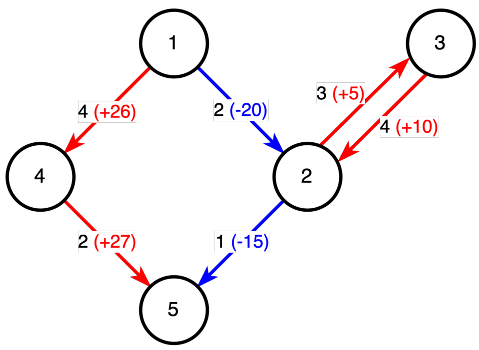

Illustration for the second example

Problem D. New Year Experiments

Time limit: 2 seconds

Ruslan gifted his friend, the hamster Artem, an array a for the New Year. Artem was very happy with the gift and decided to conduct experiments on it.

There are a total of 5 types of operations:

- **AND** — operation: a bitwise AND with the number c is applied to all elements of the array a. That is, each element a_i is replaced with a_i AND c.
- **OR** — operation: a bitwise OR with the number c is applied to all elements of the array a. That is, each element a_i is replaced with a_i OR c.
- **XOR** — operation: a bitwise XOR with the number c is applied to all elements of the array a. That is, each element a_i is replaced with a_i XOR c.
- Query for the value of the i-th element of the array.
- Query for the value of the k-th largest element of the array.

Input

Each test consists of several test cases. The first line contains a single integer t ($1 \le t \le 10^5$) — the number of test cases. The following describes the test cases.

The first line contains an integer n ($1 \le n \le 2 \cdot 10^5$) — the number of elements in the array a.

The second line contains n integers a_i ($0 \le a_i < 2^{30}$) — the elements of the array a.

The third line contains an integer q ($1 \le q \le 2 \cdot 10^5$) — the number of queries.

The next q lines contain queries in the format x c, where x — the type of operation (from 1 to 5), and c — an integer, for operations 1–3 the constraint is $0 \le c < 2^{30}$, and for operations 4–5 the constraint is $1 \le i, k \le n$.

Description of operations:

1 c : apply the AND operation to each element of the array a.

2 c : apply the OR operation to each element of the array a.

3 c : apply the XOR operation to each element of the array a.

4 i : output the value of the i-th element of the array (1-based indexing).

5 k : output the value of the k-th largest element of the array (the 1st largest is the smallest element).

It is guaranteed that the sum of all n and q does not exceed $4 \cdot 10^5$.

Output

For each query of type 4 i or 5 k, output the corresponding value on a separate line:

- For a query of type 4 i, output the value of the i-th element of the array (1-based indexing).
- For a query of type 5 k, output the value of the k-th largest element of the array (the 1st largest is the smallest element).

Scoring

Points for each subtask are awarded only if all tests for that subtask and the necessary subtasks are successfully passed.

Subtask	Points	Constraints	Necessary Subtasks	Comments
0	0	–		Tests from the statement.
1	10	$n, q \leq 10^3$, $t \leq 5$, all types of operations		
2	15	$n \cdot q \leq 10^8$, $t \leq 2$, all types of operations	0, 1	
3	15	$q \leq 4 \cdot 10^4$, $t \leq 5$, $\max a_i < 2^{10}$	–	
4	10	no operation of type 5	–	
5	20	$n, q \leq 2 \cdot 10^5$, no operation of type 3	–	
6	10	$n, q \leq 2 \cdot 10^5$, no operations of type 1 and 2	–	
7	20	–	0 – 6	

Example

standard input	standard output
1 3 1 2 3 3 3 1 4 2 5 1	3 0

Note

Consider the input data set from the example. Before executing the queries: $a = [1, 2, 3]$.

- 3 1: apply the XOR operation to each element of the array a with the number 1: $a = [0, 3, 2]$.
- 4 2: output the value of the 2-nd element of the array, the 2-nd element is equal to 3.
- 5 1: output the value of the 1-st largest element of the array, in this case, it is 0.

Bitwise AND (**AND**) is a binary operation on a pair of non-negative integers. To compute the bitwise AND of two numbers, one must consider the representation of both numbers in binary. The result is a number whose binary representation has a one in each position if there is a one in the binary representation of both arguments.

Bitwise OR (**OR**) is a binary operation on a pair of non-negative integers. To compute the bitwise OR of two numbers, one must consider the representation of both numbers in binary. The result is a number whose binary representation has a one in each position if there is a one in the binary representation of at least one of the arguments.

Bitwise exclusive OR (**XOR**) is a binary operation on a pair of non-negative integers. To compute the bitwise exclusive OR of two numbers, one must consider the representation of both numbers in binary. The result is a number whose binary representation has a one in each position if there is a one in the binary representation of exactly one of the arguments.

Problem E. Planet Parade

Time limit: 3 seconds

You are given an array $a_1, a_2, \ldots, a_n$.

Count the number of permutations $p_1, p_2, \ldots, p_n$ such that the following condition holds:

$$(a_{p_1} \bmod a_{p_2}) + (a_{p_2} \bmod a_{p_3}) + \ldots + (a_{p_{n-1}} \bmod a_{p_n}) \leq \min(a_1, a_2, \ldots, a_n) + \alpha,$$

where mod denotes the modulo operation, and $\min(a_1, a_2, \ldots, a_n)$ is the minimum of the numbers $a_1, a_2, \ldots, a_n$.

Since the answer can be quite large, output it modulo $10^9 + 7$.

Recall that an array $p_1, p_2, \ldots, p_n$ is called a permutation if for each number x from 1 to n, there exists exactly one i from 1 to n such that $p_i = x$.

Input

The first line of input contains two integers n, α ($2 \leq n \leq 300\,000$, $|\alpha| \leq 1$) — the length of the array a and the parameter α.

The second line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 10^9$) — the elements of the array a.

Output

In a single line, output the answer to the problem — the number of permutations that satisfy the condition, modulo $10^9 + 7$.

Scoring

Subtask	Points	Additional constraints			Necessary Groups	Comments
		n	α	a_i		
0	0	–	–	–	–	Tests from the statement
1	9	$n \leq 10$	–	–	–	
2	11	$n \leq 20$	$\alpha \leq 0$	all a_i are distinct	–	
3	5	–	–	$a_i \leq 2$	–	
4	8	–	–	a_i is a power of two	3	
5	7	–	$\alpha = -1$	–	–	
6	9	$n \leq 300$	$\alpha = 0$	–	–	
7	10	$n \leq 3000$	$\alpha = 0$	–	6	
8	11	–	$\alpha = 0$	–	6,7	
9	15	$n \leq 300$	–	$a_i \geq 2$	–	
10	4	$n \leq 300$	–	–	0,1,2,6,9	
11	4	–	–	$a_i \geq 2$	9	
12	7	–	–	–	0-11	

Examples

standard input	standard output
5 0 2 3 6 12 18	4
7 0 5 4 1 1 1 20 20	228
10 -1 121 12 3 4 940412847 121 3 4 12 121	48
11 1 1 1 1 1 2 2 2 2 2 2 2	18748800
16 1 1 1 1 2 4 8 16 32 64 128 256 512 1024 2048 4096 8192	28574934
3 -1 1000000000 999999999 999999998	1
20 0 604661760 120932352 46656 144 2592 3 139968 60466176 2 18 2519424 5038848 432 20155392 5184 36 72 9 1296 279936	131073
8 1 1 3 3 2 5 3 5 6	48
5 1 6 1 1 3 3	72
13 -1 1 1 1 1 1 1 1 1 1 1 1 1 1	227020758
4 1 100 51 26 14	0

Note

In the first test, the suitable arrays-permutations of the array a are:

1. $[18, 2, 12, 6, 3]$, $18 \bmod 2 + 2 \bmod 12 + 12 \bmod 6 + 6 \bmod 3 = 2 \le 2$
2. $[18, 6, 2, 12, 3]$, $18 \bmod 6 + 6 \bmod 2 + 2 \bmod 12 + 12 \bmod 3 = 2 \le 2$
3. $[12, 2, 18, 6, 3]$, $12 \bmod 2 + 2 \bmod 18 + 18 \bmod 6 + 6 \bmod 3 = 2 \le 2$
4. $[12, 6, 2, 18, 3]$, $12 \bmod 6 + 6 \bmod 2 + 2 \bmod 18 + 18 \bmod 3 = 2 \le 2$

In the fifth and seventh tests, the elements of the array are also, as in all tests, provided **in one line**, while in the statement the elements of the array are spread over several lines for better readability.